Postgres Basics

JonBakken and Vladimir Podstavkov
Fermilab Computing Division

Installation

- Use the standard distribution from the official web site:
  http://www.postgresql.org/download/linux

- We use these packages for USCMS postgres:
  - postgresql-8.3.5-1PGDG.rhel4
  - postgresql-devel-8.3.5-1PGDG.rhel4
  - postgresql-server-8.3.5-1PGDG.rhe4
  - postgresql-libs-8.3.5-1PGDG.rhel4

- Considerable performance improvements were noticed going from postgres rev 8.1 to 8.3. Everyone should consider using this rev.
  - Use the normal dump and restore technique to upgrade

For USCMS, we accept all the defaults from the standard installation, except those documented on the following pages.

Post-Installation

Create the file /etc/sysconfig/pgsql/postgresql something like this:

```
# Database location
PGDATA=/diskb/pnfs/db/psql-data - preferably not on the system disk
# Additional options
PGOPTS=-i
```

- This is a very convenient way to define the database location and to add default options
- Use separate disks for the database files to reduce unnecessary IO!

Parameters

shared_buffers = 32MB - in postgresql.conf

- Definitely not enough for serious load.   At least 256MB recommended
- If you change this parameter don't forget to change the kernel configuration accordingly!   Add something like:
  - kernel.shmmax=536870912 in /etc/sysctl.conf file
  - kernel.shmall=536870912   in /etc/sysctl.conf file
  - To change the actual values without rebooting the node do:

```
echo 536870912 > /proc/sys/kernel/shmmax
echo 536870912 > /proc/sys/kernel/shmall
```

work_mem = 1MB - in postgresql.conf

- Increase to 16MB or even more, depends on queries.
- For the USCMS replica manager it's set to 512MB

Parameters

# Free Space Map - in postgresql.conf

max_fsm_pages = 204800

- Increase to 1000000 or so...
- 1,000,000 allows to handle 8 GB of free space in the database.

# Archiving - in postgresql.conf

archive_mode = on

- Allows archiving to be done - this is done continuously

archive_command = '/home/enstore/P/p8backup/wal_backup.sh %p %f'

- You furnish the script that is executed to archive a logfile segment. Our script copies the writeahead log file (the transaction log) to another node
- You don't need slony for backups if you use archives

# Logging - in postgresql.conf

log_directory = '/diska/postgres-log'

- Directory where log files are written, different drive is preferable.

log_line_prefix = '<%d> <%t> ' (database name and time)

- Useful to make log messages more informative

pg_hba.conf

```
    ............
host    all        all        127.0.0.1/32        md5
............
```

- Kerberos is the best choice if available.
- If not - md5 recommended for extra security
  - Allows you to create the database users with an encrypted password
  - Place the password(s) in ~/.pgpass file
  - # ls -l ~/.pgpass   (note file permissions)

    ```
    -rw-------   1 root root 1732 Nov 20 15:00 /root/.pgpass
    ```
  - # head ~/.pgpass   (X's are password)

    ```
    #hostname:port:database:username:password
    localhost:5432:dcachedb:dbuser:XXXXXXXXXX
    cmsnode01:5432:dcachedb:dbuser:XXXXXXXXXX
     .................. (and so on)
    ```
- The pgpass method is fully support in dCache and USCMS has used it for years without problems

Backups

USCMS does full backups twice a day using cron job. Rough procedure is:

```
# Tell the postgres we are going to do a backup
psql -U dbuser template1 -c "checkpoint; select pg_start_backup('${dest}.tar.gz')" > /dev/null

# Create the tarball:
(cd ${fullpath}/..; tar cfz ${dest}.tar.gz --exclude ./psql-data/pg_xlog .)
if [ "$?" != "0" ] ; then
  echo "Tar cfz ${dest}.tar.gz returned error. Some files might be modified in backup. Ignored."
fi

# Tell the postgres we are done
psql -U dbuser template1 -c "select pg_stop_backup()" > /dev/null

# Copy the tarball to a remote server to save.
rsync -a  ........ or scp .............
```

USCMS backups are monitored by their zabbix monitoring framework.

Vacuuming

In postgres 8.3, auto-vacuuming is enabled by default.

- We leave this parameter enabled
- We do not do any vacuuming by cronjobs or on a manual basis

Prior to 8.3, it was necessary to enable auto-vacuuming in 3 places in the postgresql.conf file:

- stats_start_collector = true
- stats_row_level = true
- autovacuum = on

1 USCMS Specific change

We run our postgres application as user enstore, not the default user postgres

- This requires us to change the startup scripts to use -U enstore everywhere
- Our databases are owned by user enstore

User enstore has kerberos credentials that allow easy copying of files between servers at FNAL, and user postgres does not have these credentials.

The manual does indicate it is a good idea not to use the default postgres user to reduce hacking vulnerabilities.

USCMS dCache Databases

We run the following databases to support USCMS dCache, each on a separate high-end node

- PNFS - this is a cluster of 26 separate postgres databases on 1 node
    - We are experimenting with moving PNFS+Companion to Berkeley for a throughput improvement of about 4.
- Companion - also runs on the PNFS server node
- SRM/Pin
- Replica Manager - 1 database for each replica manager (we have 2)
- Billing
- Monitoring
    - We've found it beneficial to record results in a local database rather than continually querying the primary databases.
        - For example, we store the pnfsid/filename translation, the crc, pool filelist. This has improved performance considerably.
    - We also store quota information in the database.

For SRMWatch, special access rights need to be applied to the SRM databases

- Should be documented in the SRMWatch rpm (See Dmitry for details)

For Billing plots, we provide OpenLaszlo application. (See Vladimir for details)